On the Minimum Cost Range Assignment Problem

Paz Carmi*and Lilach Chaitman-Yerushalmi †

February 17, 2015

Abstract

We study the problem of assigning transmission ranges to radio stations placed arbitrarily in a d-dimensional (d-D) Euclidean space in order to achieve a strongly connected communication network with minimum total power consumption. The power required for transmitting in range r is proportional to r^α, where α is typically between 1 and 6, depending on various environmental factors. While this problem can solved optimally in 1D, in higher dimensions it is known to be NP-hard for any $\alpha \geq 1$.

For the 1D version of the problem, i.e., radio stations located on a line and $\alpha \geq 1$, we propose an optimal $O(n^2)$-time algorithm. This improves the running time of the best known algorithm by a factor of n. Moreover, we show a polynomial-time algorithm for finding the minimum cost range assignment in 1D whose induced communication graph is a t-spanner, for any $t \geq 1$.

In higher dimensions, finding the optimal range assignment is NP-hard; however, it can be approximated within a constant factor. The best known approximation ratio is for the case $\alpha = 1$, where the approximation ratio is 1.5. We show a new approximation algorithm with improved approximation ratio of $1.5 - \epsilon$, where $\epsilon > 0$ is a small constant.

1 Introduction

A wireless ad-hoc network is a self-organized decentralized network that consists of independent radio transceivers (transmitter/receiver) and does not rely on any existing infrastructure. The network nodes (stations) communicate over radio channels. Each node broadcasts a signal over a fixed range and any node within this transmission range receives the signal. Communication with nodes outside the transmission range is done using multi-hops, i.e., intermediate nodes pass the message forward and form a communication path from the source node to the desired target node. The twenty-first century witnesses widespread deployment of wireless networks for professional and private applications. The field of wireless communication continues to experience unprecedented market growth. For a comprehensive survey of this field see [7].

Let S be a set of points in the d-dimensional Euclidean space representing radio stations. A *range assignment* for S is a function $\rho : S \to \mathbb{R}^+$ that assigns each point a transmission range (radius). The cost of a range assignment, representing the power consumption of the network, is defined as $cost(\rho) = \sum_{v \in S} (\rho(v))^\alpha$ for some real constant $\alpha \geq 1$, where α varies between 1 and values higher than 6, depending on different environmental factors [7].

A range assignment ρ induces a *directed communication graph* $G_\rho = (S, E_\rho)$, where $E_\rho = \{(u, v) : \rho(u) \geq |uv|\}$ and $|uv|$ denotes the Euclidean distance between u and v. A range assignment ρ is *valid* if the induced (communication) graph G_ρ is strongly connected. For ease of presentation, throughout the paper we refer to the terms 'assigning a range $|uv|$ to a point $u \in S$' and 'adding a directed edge (u, v)' as equivalent.

We consider the d-D Minimum Cost Range Assignment (MinRange) problem, that takes as input a set S of n points in $\mathbb{R}^d$, and whose objective is finding a valid *range assignment* for S of minimum cost. This problem has been considered extensively in various settings, for different values of d and α, with additional requirements and modifications. Some of these works are mentioned in this section.

*Department of Computer Science, Ben-Gurion University of the Negev, Israel
†Department of Computer Science, Ben-Gurion University of the Negev, Israel

In [6], Kirousis et al. considered the 1D MinRange problem (the radio stations are placed arbitrarily on a line) and showed an $O(n^4)$-time algorithm which computes an optimal solution for the problem. Later, Das et al. [5] improved the running time to $O(n^3)$. Here, we propose an $O(n^2)$-time exact algorithm, this improves the running time of the best known algorithm by a factor of n without increasing the space complexity. The novelty of our method lies in separating the range assignment into two, *left* and *right*, range assignments (elaborated in Section 2). This counter intuitive approach allows us to achieve the aforementioned result and, moreover, to compute an optimal range assignment in 1D with the additional requirement that the induced graph is a t-spanner, for a given $t \geq 1$.

A directed graph $G = (S, E)$ is a t-spanner for a set S, if for every two points $u, v \in S$ there exists a path in G from u to v of length at most $t|uv|$. The importance of avoiding flooding the network when routing, was one of the reasons that led researchers to consider the combination of range assignment and t-spanners, e.g., [1, 8, 9, 10], as well as the combination of range assignment and hop-spanners, e.g., [4, 6]. While bounded-hop spanners bound the number of intermediate nodes forwarding a message, t-spanners bound the relative distance a message is forward. For the 1D bounded-hop range assignment problem, Clementi et al. [4] showed a 2-approximation algorithm whose running time is $O(hn^3)$. To the best of our knowledge, we are the first to show an algorithm that computes an optimal solution for the range assignment with the additional requirement that the induced graph is a t-spanner.

While the 1D version of the MinRange problem can be solved optimally, for any $d \geq 2$ and $\alpha \geq 1$, it has been proven to be NP-hard (in [6] for $d \geq 3$ and $1 \leq \alpha < 2$ and later in [3] for $d \geq 2$ and $\alpha > 1$). However, some versions can be approximated within a constant factor. For $\alpha = 2$ and any $d \geq 2$ Kirousis et al. [6] gave a 2-approximation algorithm based on the minimum spanning tree (although they addressed the case of $d \in \{2, 3\}$ their result holds for any $d \geq 2$). The best known approximation ratio is for the case $\alpha = 1$, where the approximation ratio is 1.5 [2]. We show a new approximation algorithm for this case[1] with improved approximation ratio of $1.5 - \epsilon$, for a suitable constant $\epsilon > 0$. We do not focus on increasing ϵ but rather on showing that there exists an approximation ratio for this problem that is strictly less than 1.5. This is in contrast to classic problems, such as metric TSP and strongly connected sub-graph problems, for which the 1.5 ratio bound has not yet been breached.

2 Minimum Cost Range Assignment in 1D

In the 1D version of the MinRange problem, the input set $S = \{v_1, ..., v_n\}$ consists of points located on a line. For simplicity, we assume that the line is horizontal and for every $i < j$, v_i is to the left of v_j. Given two indices $1 \leq i < j \leq n$, we denote by $S_{i,j}$ the subset $\{v_i, ..., v_j\} \subseteq S$.

We present two polynomial-time algorithms for finding optimal range assignments, the first, in Section 2.1, for the basic 1D MinRange problem, and the second, in Section 2.2, subject to the additional requirement that the induced graph is a t-spanner (the 1D MinRangeSpanner problem). Our new approach for solving these problems requires introducing a variant of the *range assignment*. Instead of assigning each point in S a radius, we assign each point two directional ranges, *left range assignment*, $\rho^l : S \rightarrow \mathbb{R}^+$, and *right range assignment*, $\rho^r : S \rightarrow \mathbb{R}^+$. A pair of assignments (ρ^l, ρ^r) is called a *left-right assignment*. Assigning a point $v \in S$ a left range $\rho^l(v)$ and a right range $\rho^r(v)$ implies that in the induced graph, $G_{\rho^{lr}}$, v can reach every point to its left up to distance $\rho^l(v)$ and every point to its right up to distance $\rho^r(v)$. That is, $G_{\rho^{lr}}$, contains the directed edge (v_i, v_j) iff one of the following holds: (i) $i < j$ and $|v_i v_j| \leq \rho^r(v_i)$, or (ii) $j < i$ and $|v_i v_j| \leq \rho^l(v_i)$. The cost of an assignment (ρ^l, ρ^r), is defined as $cost(\rho^l, \rho^r) = \sum_{v \in S}(\max\{\rho^l(v), \rho^r(v)\})^\alpha$.

Our algorithms find a *left-right assignment* of minimum cost that can be converted into a *range assignment* ρ with the same cost by assigning each point $v \in S$ a range $\rho(v) = \max\{\rho^l(v), \rho^r(v)\}$. Note that any valid *range assignment* for S can be converted to a a *left-right assignment* with the same cost, by assigning every point $v \in S$, $\rho^l(v) = \rho^r(v) = \rho(v)$. To be more precise, either $\rho^l(v)$ or $\rho^r(v)$ should be reduced to $|vu|$, where u is the farthest point in the directional range (for Lemma 1 to hold). Therefore, a minimum cost *left-right assignment*, implies a minimum cost *range assignment*.

In addition to the *cost* function, we define $cost'(\rho^l, \rho^r) = \sum_{v \in S}((\rho^l(v))^\alpha + (\rho^r(v))^\alpha)$, and refine the term of *optimal solution* to include only solutions that minimize $cost'(\rho^l, \rho^r)$ among all solutions, (ρ^l, ρ^r), with minimum $cost(\rho^l, \rho^r)$.

[1] Values of α smaller than 2 correspond to areas, such as, corridors and large open indoor areas [7].

2.1 An Optimal Algorithm for the 1D MinRange Problem

Das et al. [5] state three basic lemmas regarding properties of an optimal range assignment. The following three lemmas are adjusted versions of these lemmas for a *left-right assignment*.

Lemma 1. *In an optimal solution (ρ^l, ρ^r) for every $v_i \in S$, either $\rho^l(v_i) = 0$ or $\rho^l(v_i) = |v_i v_j|$ and similarly, either $\rho^r(v_i) = 0$ or $\rho^r(v_i) = |v_i v_k|$ for some $j \leq i \leq k$.*

Lemma 2. *Given three indices $1 \leq i < j < k \leq n$, consider an optimal solution for $S_{i,k}$, denoted by (ρ^l, ρ^r), subject to the condition that $\rho^l(v_j) \geq |v_i v_j|$ and $\rho^r(v_j) \geq |v_j v_k|$, then,*
- *for all $m = i, ..., j-1$, $\rho^r(v_m) = |v_m v_{m+1}|$ and $\rho^l(v_m) = 0$; and*
- *for all $m = j+1, ..., k$, $\rho^l(v_m) = |v_m v_{m-1}|$ and $\rho^r(v_m) = 0$.*

Lemma 3. *In an optimal solution (ρ^l, ρ^r), $\rho^l(v_1) = 0$ and $\rho^r(v_1) = |v_1 v_2|$.*

Lemma 1 allows us to simplify the notation $\rho^x(v_i) = |v_i v_j|$ for $x \in \{l, r\}$ and $1 \leq i, j \leq n$, and write $\rho^x(i) = j$ for short. We solve the MinRange problem using dynamic programming. Given $1 \leq i < n$, we denote by $OPT(i)$ the cost of an optimal solution for the sub-problem defined by the input $S_{i,n}$, subject to the condition that $\rho^r(i) = i + 1$. Note that the cost of an optimal solution for the whole problem is $OPT(1)$.

In Section 2.1.1 we present an algorithm with $O(n^3)$ running time and $O(n^2)$ space (the same time and space as in [5]). Then, in Section 2.1.2 we reduce the running time to $O(n^2)$.

2.1.1 A Cubic-Time Algorithm

Algorithm 1DMinRA (Algorithm 1) applies dynamic programming to compute the values $OPT(i)$ for every $1 \leq i \leq n$ and store them in a table, T. Finally, it outputs the value $T[1]$. In our computation we use a 2-dimensional matrix, *Sum*, storing for every $1 \leq i < j \leq n$ the sum $\sum_{m=i}^{j-1} |v_m v_{m+1}|^\alpha$. While

Algorithm 1 1DMinRA(S)

for $i = n-1$ **downto** 1 **do**
  for $j = n$ **downto** $i+1$ **do**

$$\text{Sum}[i,j] \leftarrow \begin{cases} |v_{n-1}v_n|^\alpha & , i = n-1 \\ \text{Sum}[i+1,n] + |v_i v_{i+1}|^\alpha & , j = n \\ \text{Sum}[i,j+1] - |v_j v_{j+1}|^\alpha & , \text{otherwise} \end{cases}$$

for $i = n-1$ **downto** 1 **do**

$$\text{T}[i] \leftarrow \begin{cases} 2|v_i v_{i+1}|^\alpha & , i = n-1 \\ \min\limits_{\substack{i<k<n \\ k<k'\leq n}} \{\text{Sum}[i,k'-1] + \text{T}[k'-1] - |v_{k'-1}v_{k'}|^\alpha \\ \quad + \max\{|v_i v_k|^\alpha, |v_k v_{k'}|^\alpha\}\} & , \textit{otherwise} \end{cases}$$

return T[1]

the table T maintains only the costs of the solutions, the optimal assignment can be easily retrieved by backtracking the cells leaded to the optimal cost and assigning the associated ranges (described in the proof of Lemma 4).

Correctness. We prove that for every $1 \leq i \leq n$, the value assigned to cell $T[i]$ by the algorithm equals $OPT(i)$. Trivially, $OPT(n-1)$ indeed equals $2|v_i v_{i+1}|^\alpha$. Assume, during the i-th iteration it holds that $T[i'] = OPT(i')$ for every $i < i' < n$, the correctness of the computation done during the i-th iteration is given in Lemma 4.

Lemma 4. *Given an index i with $1 \leq i \leq n-1$,*

$$OPT(i) = \min_{\substack{i<k<n \\ k<k'\leq n}} \left\{ \sum_{m=i}^{k'-2} |v_m v_{m+1}|^\alpha + OPT(k'-1) - |v_{k'-1}v_{k'}|^\alpha + \max\{|v_i v_k|^\alpha, |v_k v_{k'}|^\alpha\} \right\}.$$

Proof. Let X_i denote the right side of the equation, we prove $OPT(i) = X_i$.

$OPT(i) \leq X_i$: We show that all costs that appear as min function arguments in X_i correspond to valid assignments and thus infer, by the optimality of $OPT(i)$, that the above inequality holds. Consider an argument with parameters k and k'. We associate it with an assignment (ρ^l, ρ^r) defined as follows (see Fig. 1(a)). For $m \geq k'-1$ the assignment is inductively defined by $OPT(k'-1)$. For every $i \leq m < k$, $\rho^l(m) = m$ and $\rho^r(m) = m+1$, for every $k < m < k'$, $\rho^l(m) = m-1$ and $\rho^r(m) = m$ ($v_{k'-1}$ is reassigned) and for k, $\rho^l(k) = i$, $\rho^r(k) = k'$. By the validity of $OPT(k'-1)$, every two points among $S_{k'-1,n}$ are (strongly) connected. Our assignment for $S_{i,k'-1}$ guarantees the connectivity between every two points in $S_{i,k'-1}$, and thus between every two points in $S_{i,n}$.

$OPT(i) \geq X_i$: Consider an optimal solution (ρ^l, ρ^r) for the points $S_{i,n}$ subject to the condition that $\rho^r(i) = i+1$. Let v_k be a point to the right of v_i with $\rho^l(k) = i$ and let $\rho^r(k) = k'$. Note that since v_i is the leftmost point and the induced graph is strongly connected, such a point necessarily exists.
Next we show that there is no edge directed either left or right connecting two points on different sides of $v_{k'}$ in $G_{\rho^{lr}}$, except for possibly an edge $(v_j, v_{k'-1})$ with $j > k'$. Assume towards contradiction that the former does not hold, i.e., there exists $i < t < k'$, with $\rho^r(t) \geq k'$; then, reassigning $\rho^r(k) = \max\{t, k\}$ maintains the connectivity, and reduces the value of $cost'$ without increasing the value of $cost$ in contradiction to the optimality of the solution. Now, let v_j be a point to the right of $v_{k'}$ with $\rho^l(j) = j' \in [i, k']$, we show that $j' \geq k'-1$. Consider a point $j' < t < k'$, as we have shown, $\rho^r(t) < k'$. By symmetric arguments we have $\rho^l(t) > j'$ (see Fig. 1(b)). Namely, there is no edge going out of the interval $(v_{j'}, v_{k'})$. Thus, connectivity can be achieved only if this interval is empty of vertices, i.e., either $j' = k'-1$ or $j' = k'$ (note that $k'-1 > i$).
The above observation allows us to divide the problem into two independent subproblems, one for the points $S_{i,k'-1}$ subject to the constraints $\rho^l(k) = i$ and $\rho^r(k) = k'$, and the other for the points $S_{k'-1,n}$ subject to the artificial constraint $\rho^r(k'-1) = k'$ that guarantees the existence of a path from $k'-1$ to k', due to the solution of the first subproblem, but should not be paid for. Regarding the first subproblem, by Lemma 2, in an optimal assignment, for every $i \leq m < k$, $\rho^l(m) = m$ and $\rho^r(m) = m+1$, and for every $k < m \leq k'-1$, $\rho^l(m) = m-1$ and $\rho^r(m) = m$. Thus, its cost is $\sum_{m=i}^{k'-2} |v_m v_{m+1}|^\alpha + \max\{|v_k v_i|^\alpha, |v_k v_{k'}|^\alpha\}$. The cost of an optimal solution to the second subproblem is $OPT(k'-1) - |v_{k'-1} v'_k|^\alpha$. Hence, the $cost$ of an optimal solution to the whole problem is the sum of the above costs and the lemma follows.

□

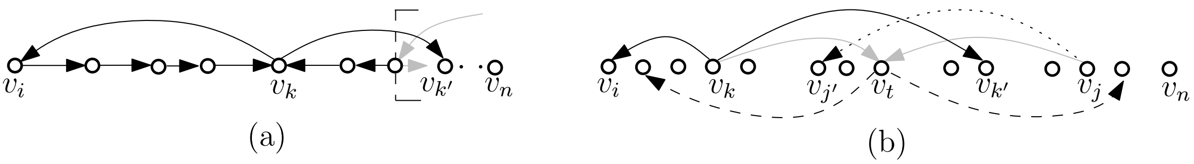

Figure 1: (a) An illustration of the assignment associated with $OPT(i)$ with respect to given parameters k and k'. In gray are range assignments associated with $OPT(k'-1)$. (b) An illustration of the proof of Lemma 4. In dashed arrows, the impossible ranges of v_t and in gray, the alternative assignment of lower $cost'$.

Complexity. Obviously, Algorithm 1DMinRA requires $O(n^2)$ space. Regarding the running time, $O(n)$ iterations are performed during the algorithm, each iteration takes $O(n^2)$ time Therefore, the total running time is $O(n^3)$ and Lemma 5 follows.

Lemma 5. *Algorithm* 1DMinRA *runs in $O(n^3)$ time using $O(n^2)$ space.*

2.1.2 A Quadratic-Time Algorithm

In this section we consider Algorithm 1DMinRA from previous section and reduce its running time to $O(n^2)$. Consider the equality stated in Lemma 4. Observe that given fixed values i and k', the value

k that minimizes the argument of the min function with respect to i and k' is simply the value k that minimizes $\max\{|v_i v_k|^\alpha, |v_k v_{k'}|^\alpha\}$. This value is simply the closest point to the midpoint of the segment $\overline{v_i v_{k'}}$, denoted by $c(i, k')$. Thus,

$$OPT(i) = \min_{i+1<k'\leq n} \left\{ \begin{array}{l} \displaystyle\sum_{m=i}^{k'-2} |v_m v_{m+1}|^\alpha + OPT(k'-1) - |v_{k'-1} v_{k'}|^\alpha \\ \qquad\qquad\qquad + \max\{|v_i v_{c(i,k')}|^\alpha, |v_{c(i,k')} v_{k'}|^\alpha\} \end{array} \right\}.$$

Consider Algorithm 1DMinRA after applying the above modification in the computation of $T[i]$. Since there are only $O(n)$ sub-problems to compute, each in $O(n)$ time, the running time reduces to $O(n^2)$ and the following theorem follows.

Theorem 6. *The 1D* MinRange *problem can be solved in $O(n^2)$ time using $O(n^2)$ space.*

2.2 An Optimal Algorithm for the 1D MinRangeSpanner Problem

Given a set $S = \{v_1, .., v_n\}$ of points in $1D$ and a value $t \geq 1$, the 1D MinRangeSpanner problem aims to find a minimum cost range assignment for S, subject to the requirement that the induced graph is a t-spanner. We present a polynomial-time algorithm which solves this problem optimally and follows the same guidelines as Algorithm 1DMinRA.

We begin with providing the key notions required for understanding the correctness of the algorithm, followed by its description. Due to space limitation, we do not supply a formal proof. The first and most significant observation, is that the problem can still be divided into two subproblems in the same way as in Algorithm 1DMinRA, by similar arguments to those of Lemma 4. In Lemma 4 we show that any assignment that does not satisfy the conditions required for the division can be adjusted to a new assignment with a lower value of $cost'$ that preserves connectivity. The new assignment, however, preserves also the lengths of the shortest paths, which make the argument legitimate for this problem as well.

The two problems (MinRange and MinRangeSpanner) differ when it comes to solving each of the above subproblems. Consider the left subproblem, i.e., of the form described in Lemma 2. The optimal assignment for it is no longer necessarily the one stated in the lemma, since it does not ensure the existence of t-spanning paths. Therefore, our algorithm divides problems of this form into smaller subproblems handled recursively (see Fig. 2, right). Dealing with such subproblems, requires defining new parameters: a rightmost input point v_j, and the length of the shortest paths connecting v_i to v_j, v_j to v_i and v_i to v_{i+1} not involving points in $S_{i,j}$ except for the endpoints, denoted by $\overrightarrow{\delta}$, $\overleftarrow{\delta}$, and δ^i , respectively. Regarding the computation of a subproblem, since points may be covered now by vertices outside the subproblem domain, we allow v_k to have either a right or a left range equals 0 (in the terms of Algorithm 1DMinRA, either $k = i$ or $k = k'$).

Another key observation is that any directed graph G over S is a t-spanner for S iff for every $1 \leq i < n$ there exists a t-spanning path from v_i to v_{i+1} and from v_{i+1} to v_i. Moreover, given that G is strongly connected implies that the addition of an edge between consecutive points does not effect the length of the shortest path between any other pair of consecutive points. Therefore, for subproblems with $j = i+1$ we assign $\rho^r(i) = i+1$ (resp. $\rho^l(i+1) = i$) iff $\overrightarrow{\delta}/|i, i+1| > t$ (resp. $\overleftarrow{\delta}/|i, i+1| > t$) and thus ensuring that the induced graph is a t-spanner.

Our algorithm may consider solutions in which an assignment to a node is charged more than once in the total cost; however, for every such solution, there exists an equivalent one in which the charging is done properly and is preferred by the algorithm due its lower cost.

We denote by $OPT(i, j, \overrightarrow{\delta}, \overleftarrow{\delta}, \delta^i)$ the cost of an optimal solution to the sub-problem defined by the input $S_{i,j}$ subject to the parameters $\overrightarrow{\delta}$, $\overleftarrow{\delta}$, and δ^i representing the lengths of the shortest external paths as defined earlier in this section. Let $\Delta_{i,j} = \{2|v_l v_i| + |v_i v_j|, 2|v_j v_r| + |v_i v_j| : l \leq i < j \leq r\}$. We compute $OPT(i, j, \overrightarrow{\delta}, \overleftarrow{\delta}, \delta^i)$ for every $1 \leq i < j \leq n$, and the corresponding $\overrightarrow{\delta}, \overleftarrow{\delta}, \delta^i \in \Delta_{i,j}$ iteratively, while in stage x all subproblems with $j - i = x$ are solved. The computation is derived from the equalities below. For simplicity of presentation, we overload notation and write $|i, j|$ to mean $|v_i v_j|$. In addition, we write

$\varnothing$ in place of δ^i where $\delta^i = \overrightarrow{\delta}$.

$$OPT(i, i+1, \overrightarrow{\delta}, \overleftarrow{\delta}, \delta^i) = \overrightarrow{r} + \overleftarrow{r}, \text{where}$$
$$\overrightarrow{r} = \begin{cases} |i, i+1| \; (*assigning\ \rho^r(i) = i+1*) & , \overrightarrow{\delta}/|i, i+1| > t \\ 0 & , otherwise \end{cases}$$

and $\overleftarrow{r}$ is defined symmetrically. For $j > 1$, we have

$$OPT(i, j, \overrightarrow{\delta}, \overleftarrow{\delta}, \delta^i) =$$
$$\min_{\substack{i \le k \le j \\ k \le k' \le j}} \begin{cases} |i, i+1|^\alpha + OPT(i,\, i+1,\, |i, i+1|,\, |i+i, j| + \overleftarrow{\delta},\, \varnothing) \\ \qquad + OPT(i+1,\, j,\, \infty,\, \overleftarrow{\delta} - |i+1, i|,\, \varnothing) & , i = k \\ |i, i+1|^\alpha + OPT(i\, i+1,\, \delta^i,\, |i, i+1|,\, \varnothing) \\ \qquad + OPT(i+1,\, j,\, |i, i+1| + \overrightarrow{\delta},\, \overleftarrow{\delta} - |i, i+1|,\, \varnothing) & , k = k' \\ \max\{|i, k|^\alpha, |k, k'|^\alpha\} \\ + OPT(i,\, i+1,\, \delta^i,\, |i+1, k| + |k, i|,\, \varnothing) \\ + OPT(i+1,\, k,\, \infty,\, |k, i+1|,\, \infty) & , i \neq k \neq k' \\ + OPT(k,\, k'-1,\, |k, k'-1|,\, \infty,\, |k, k+1|) \\ + OPT(k'-1,\, j,\, |k'-1, i| + \overrightarrow{\delta},\, \overleftarrow{\delta} - |i, k'-1|,\, |k'-1, k| + |k, k'|). \end{cases}$$

We permit either $i = k$ and then $k' = i+1$ or $k = k'$ and then $k' = i+1$ but not both.

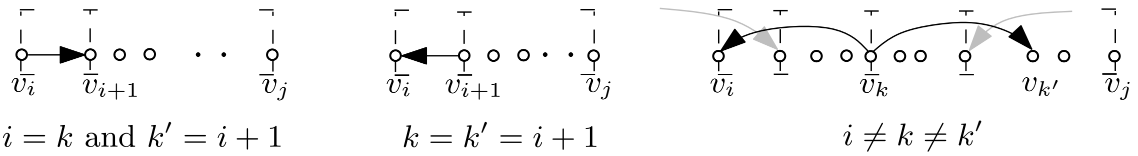

Figure 2: An illustration of the algorithm for the MinRangeSpanner problem. The ranges are illustrated in black arrows and the division to subproblems in sashed lines.

Complexity. Let Δ be the set of all distinct distances in S, then for every $v_i, v_j \in S$, $|\Delta_{i,j}| = |\Delta| = O(n)$. We fill a table with $O(n^2|\Delta|^3)$ cells, each cell is computed in $O(n^2)$ time, thus, the total running time is $O(n^4|\Delta|^3)$. As we have focused on presenting a simple and intuitive solution, rather than reducing the running time, a more careful analysis achieves a better bound on the time complexity. For example, the relevant domain of $\overrightarrow{\delta}, \overleftarrow{\delta}$, and δ^i can be estimated more precisely with respect to t. Moreover, Observation 7 allows reducing the running time by a factor of n. This is done by decreasing the number of relevant combinations of i and k' that have to be checked by the algorithm, for fixed indices j and k with $i < k < k' < j$, to $O(n)$, using similar arguments to those in Lemma 4 (see Fig. 3).

Observation 7. *Consider an optimal assignment (ρ^l, ρ^r) and a point $v_k \in S$. Let $\rho^l(k) = i$ and let k^i denote the minimal index with $k < k^i$ and $|v_k v_{k^i}| \geq |v_k v_i|$, then $k^{i+1} \leq \rho^r(k) \leq k^i$.*

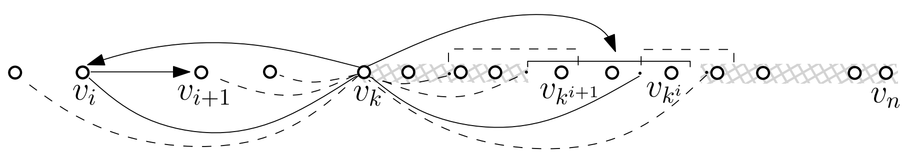

Figure 3: An illustration of Observation 7. Every pair of symmetric arcs indicates equal distances from v_k. The marked domains indicates the legal values of $\rho^r(k)$ for different values of i.

3 The MinRange Problem in Higher Dimensions

In this section we focus on the MinRange problem for dimension $d \geq 2$ and $\alpha = 1$. As all the versions of the problem for $d \geq 2$ and $\alpha \geq 1$, it is known to be NP-hard. Currently, the algorithm achieving the best approximation ratio for $\alpha = 1$ and ant $d \geq 2$ is the *Hub* algorithm with a ratio of 1.5. This algorithm was proposed by G. Calinescu, P.J.Wan, and F. Zaragoza for the general metric case, and analyzed by Ambühl et al. in [2] for the restricted Euclidean case. We show a new approximation algorithm and bound its approximation ratio from above by $1.5 - \epsilon$ for $\epsilon = 5/10^5$. Although in some cases our phrasing is restricted to the plane, all arguments hold for higher dimensions as well.

In our algorithm we use two existing algorithms, the *Hub* algorithm and the algorithm for the 1D MinRange problem introduced by Kirousis et al. [6], to which we refer as the $1D$ RA algorithm. We observe that the later algorithm outputs an optimal solution for any ordered set $V = \{v_1, ..., v_n\}$ with distance function h that satisfies the following *line alike* condition: for every $1 \leq i \leq j < k \leq l \leq n$, it holds that $h(v_i, v_l) \geq h(v_j, v_k)$. We use this algorithm for subsets of the input set that roughly lie on a line.

3.1 Our Approach

Presenting our approach requires acquaintance with the *Hub* algorithm. The *Hub* algorithm finds the minimum enclosing disk C of S centered at point $hub \in S$. Then, the algorithm sets $\rho(hub) = r_{min}$, where r_{min} is C's radius. Finally, it directs the edges of $MST(S)$ towards the hub and for each directed edge (v, u) sets $\rho(v) = |vu|$. The cost of this assignment is $w(MST(S)) + r_{min} \leq w(MST(S)) + (w(MST(S)) + w(e_M))/2$, where e_M is the longest edge in $MST(S)$ and the weight function w is defined with respect to Euclidean lengths.

To guide the reader, we give an intuition and a rough sketch of our algorithm. We characterize the instances where the *Hub* algorithm gives a better approximation than 1.5, and to generalize these cases we slightly modify it. Furthermore, we show an algorithm that prevails in the cases where the modified *Hub* algorithm fails to give an approximation ratio lower than 1.5. Before we elaborate more on the aforementioned characterization, another piece of terminology. Given a graph G over S and two points $p, q \in S$, the *stretch factor* from p to q in G is $\delta_G(p, q)/|pq|$, where $\delta_G(p, q)$ denotes the Euclidean length of the shortest path between p and q in G. We use $\sim$*large* when referring to values greater than fixed thresholds, some with respect to $w(MST(S))$, defined later.

Consider $MST(S)$ and its longest path P_M. If one of the following conditions holds, then the *Hub* algorithm or its modification results in a better constant approximation than 1.5: (A1) there exists a $\sim$*large* edge in $MST(S)$; (A2) a $\sim$*large* fraction of P_M consists of disjoint sub-paths connecting pairs of points with $\sim$*large* stretch factor, not dominated by one sub-path of at least half the fraction; or (A3) the weight $w(MST(S) \backslash P_M)$ is $\sim$*large*.

Otherwise, there are three possible cases: (B1) the graph $MST(S)$ is roughly a line; (B2) there are two points in P_M with $\sim$*large* stretch factor, i.e., there is a $\sim$*large* 'hill' in P_M, and then either $MST(S)$ roughly consists of two 1D paths; or (B3) the optimal solution uses edges connecting the two sides of the 'hill', covering $\sim$*large* fraction of it.

The last three cases are approximated using the following method. We consider every two edges connecting the two sides of the 'hill' as the edges in the optimal solution that separates the uncovered remains of the path to two independent sub-paths, i.e., not connected by an edge. (the points in both sub-paths may be connected to the middle covered area.) Note that such two edges exist. We direct the covered area to achieve a strongly connected sub-graph and solve each of the two sub-paths separately in two techniques. The first, using the $1D$ RA algorithm with a distance function implied by the input, satisfying the *line alike* condition, and applying adjustments on the output, and the second, using the *Hub* algorithm. A $(1.5 - \epsilon)$-approximation is obtained for cases (B1) and (B2), using the first technique, and for cases (B1) and (B2), using the second technique. The algorithm computed several solutions, using the aforementioned methods, and returns the one of minimum cost.

3.2 The Approximation Algorithm

The algorithm uses the following three procedures that are defined precisely at the end of the algorithm's description.

- The *flatten* procedure f - a method performing shortcuts between pairs of points on a given path P resulting in a path without two points of stretch factor greater than c_s.
- The *distance function* h_S - a distance function defined for an ordered set $P \subseteq S$, satisfying the *line alike* condition.
- The *adjustment* transformation g - a function adjusting an optimal range assignment for an ordered set $P \subseteq S$ with distance function h, to a valid assignment for P.

Let R be the forest obtained by omitting from $MST(S)$ the edges of its longest path, PM. Given a point $v \in P_M$, let $T(v)$ denote the tree of R rooted at v. For every $u \in T(v)$ let $r(u)$ denote the root of the tree in R containing u, namely, v. For a set of points $V \subset P_M$, let $T(V)$ denote the union $\bigcup_{v \in V} T(v)$. For ease of presentation, we assume the path P_M has a *left* and a *right* endpoints, thus, the *left* and *right* relations over P_M are naturally defined.

The main algorithm scheme:
Compute four solutions and return the one of minimal cost. In case of multiple assignments to a point in a solution, the maximal among the ranges counts.
<u>Solution</u> (i): apply the *Hub* algorithm.
<u>Solution</u> (ii): apply a variant of the *Hub* algorithm - find a point $c \in P_M$ that minimizes the value $r_c = \max\{|cp_1|, |cp_z|\}$, where p_1 and p_z are the endpoints of the path P_M. Assign c the range r_c, direct P_M towards r_c and bi-direct all edges in R.

(* *The rest of the algorithm handles cases (B1)-(B3) defined in Section 3.1* *)

For every edge $e \in P_M$ do :

Let P_{e^l} and P_{e^r} be the two paths of $P_M \backslash e$, to the left and to the right of e, respectively.
Apply the *flatten procedure* f on P_{e^l} and P_{e^r} to obtain the sub-paths
$P_{l'} = (p_1, p_2, ..., p_m)$ and $P_{r'} = (p_{m+1}, p_{i+2}, ..., p_z)$, respectively.
(* *Note R has been changed during the* flatten procedure *)
For every 4 points $p_l, p_{l'}, p_{r'}, p_r$ with $l \leq l' \leq m < r' \leq r$ in the flattened sub-paths:

In both solutions (iii) and (iv) direct the path $P_x = (p_l, ...p_m, p_{m+1}, ..., p_r)$ towards p_l and for each point p_i with $1 \leq i \leq z$ direct $T(p_i)$ towards p_i and assign p_i a range $w(T(p_i))$. Perform the least cost option among the following two, either add the edge (p_l, p_r), or add the two edges, one from u_l to $u_{r'}$ for $u_l \in T(p_l), u_{r'} \in T(p_{r'})$ of minimal length and the other from $u_{l'}$ to u_r for $u_{l'} \in T(p_{l'}), u_r \in T(p_r)$ of minimal length. (see illustration in Fig. 4). As for the two sub-paths $P_l = (p_1, p_2, ..., p_l)$ and $P_r = (p_r, p_{r+1}, ..., p_z)$, assign them ranges as follows:
<u>Solution</u> (iii): apply the *Hub* algorithm separately on each sub-path.
<u>Solution</u> (iv): apply the $1D$ *RA* algorithm separately on each sub-path with respect to the *distance function* h_S and perform the *transformation* g on the assignment received.

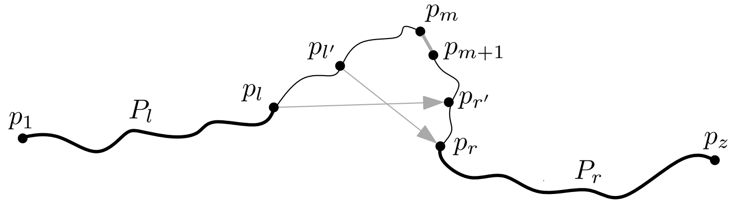

Figure 4: The two sub-paths P_l and P_r as defined in the algorithm.

The *flatten* procedure f. Let $c_s = 5/4$. Given a path $P = \{v_i, .., v_n\}$, set $Q_P = \{\}$. Let $j > i$ be the maximal index such that $\delta_P(v_i, v_j) > c_s|v_iv_j|$. If such index does not exist, let $j = i + 1$. Else ($j > i + 1$), add the edge (v_i, v_j) to P, remove the edge (v_{j-1}, v_j) from P, move the sub-path $(v_i, .., v_{j-1})$ from P to the forest R, and update $Q_P = Q_P \cup \{(v_i, v_j)\}$. Finally, repeat with the sub-path $(v_j, .., v_n)$ without initializing Q_P.

The definitions for h_S and g are given with respect to the sub-paths P_l and $P_{l'}$, the definitions for the sub-path P_r and $P_{r'}$ are symmetric.

The distance function h_S. For every two points p_j, p_k with $1 \le j \le k \le l$ we define,

$$h_S(p_j, p_k) = \min_{\substack{u \in T(p_{j'}), 1 \le j' \le j \\ v \in T(p_{k'}), k \le k' \le m}} |uv|.$$

The *adjustment* transformation g. Given an assignment $\rho' : P_l \to \mathbb{R}^+$, we transform it into an assignment $g(\rho') = \rho : P_{l'} \to \mathbb{R}^+$. First, we assign ranges a follows:

$$\rho(p_j) = \begin{cases} c_s \cdot \rho(p_j) + c_k \cdot T(p_j), 1 \le j \le l, \\ c_k \cdot T(p_j), l < j \le m, \end{cases}$$

where $c_k = 1+8(1+c_s) = 19$. The multiplicity (by c_s) handles the gaps caused by points breaking the *line alike* condition with respect to the Euclidean metric. The role of the additive part, together with the second stage of the transformation, elaborated next, is to overcome the absence of points outside the path. In the second stage, for every p_j with $1 \le j \le m$, let $1 \le j^- < j$ be the minimal index for which there exists $u \in T(p_{j^-})$ with $|p_j u| \le c_k \cdot w(T(p_j))$, and let $j < j^+ \le m$ be the maximal index for which there exists $u \in T(p_{j^+})$ with $|p_j u| \le c_k \cdot w(T(p_j))$, direct the sub-path between p_{j^-} and p_{j^+} towards p_j. See Fig. 5 for illustration.

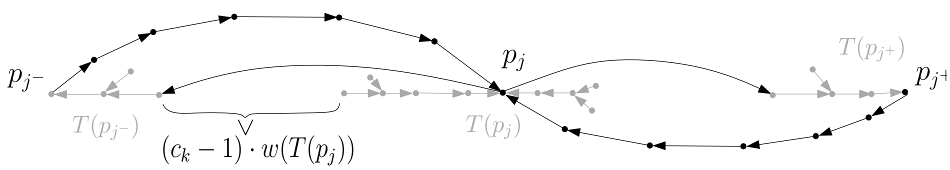

Figure 5: An illustration of the second stage in the adjustment transformation g.

The indexing of points in P_M and notations introduced in the algorithm are used throughout this section. The tree $T(p_i) \in R$, for $1 \le i \le z$, is sometimes denoted by T_i for short.

3.3 The Validity of the Output

We consider each solution separately and show it forms a *valid* assignment ρ.

Validity of Solution (i). Follows from the validity of the *Hub* algorithm

Validity of Solution (ii). The subgraph of G_ρ induced by the points of P_M is strongly connected due to the validity of the *Hub* algorithm. All trees in R are bi-directed trees sharing a common point with P_M and therefore, the whole graph, G_ρ, is strongly connected.

Validity of Solution (iii). Each tree in R induces a strongly connected subgraph of G_ρ, thus, it is suffices to show the connectivity of the minor obtained from G_ρ by contracting all trees of R. The subgraph of G_ρ induced by the points of the middle sub-path P_x forms either one directed cycle or two directed cycles sharing common vertices. By the correctness of the *Hub* algorithm, each of the sub-paths P_l and P_r induces a strongly connected subgraph of G_ρ. In addition, each of them shares a common vertex with the middle sub-path, thus, the whole graph, G_ρ, is strongly connected.

Validity of Solution (iv). Since we already verified the validity of Solution (iii), we are only left to show that each of P_l and P_r induce strongly connected subgraphs in G_ρ. We consider the sub-path P_l, while the case of P_r is symmetric.

Let $\rho_l : P_l \to \mathbb{R}^+$ denote the assignments obtained by applying the 1D algorithm on the sub-path P_l with respect to h_S. Due to the validity of ρ_l, the graph induced by ρ_l with respect to h_S is strongly connected. Let (p_i, p_j) be an edge in this graph, we show that there exists a directed path from p_i to p_j in G_ρ. Assume w.l.o.g., $1 \le i < j \le l$. By the definition of h_S, there exist $u \in T(p_{i'})$ and $v \in T(p_{j'})$ with $1 \le i' \le i < j \le j' \le m$, such that $\rho_l(p_i) \ge h_S(p_i, p_j) = |uv|$. Since the final assignment ρ is obtained after applying the adjustment transformation g, we have $\rho(p_i) \ge c_s|uv|$.

Case 1: $|uv| \le (c_k - 1) \cdot w(T(p_{i'}))$ or $|uv| \le (c_k - 1) \cdot w(T(p_{j'}))$.

Assume, w.l.o.g., that the second condition holds, then $|p_{j'} u| \le c_k \cdot w(T(p_{j'}))$ and thus, by the definition of transformation g, the directed path from $p_{i'}$ to $p_{j'}$ and the edge $(p_{j'}, u)$ are contained in G_ρ. Together with the directed path in $T(p_{i'})$ from u to the root $p_{i'}$ they form a cycle containing both p_i and p_j.

Case 2: both $|uv| > (c_k - 1) \cdot w(T(p_{i'}))$ and $|uv| > (c_k - 1) \cdot w(T(p_{j'}))$.

Since $p_{i'}, p_{j'} \in P_{l'}$, and $P_{l'}$ is the output sub-path after performing the *flatten* procedure, then $\delta_{P_{l'}}(p_{i'}, p_{j'}) \leq c_s|p_{i'}p_{j'}|$. Therefore, we have

$$\begin{aligned}
|p_i p_j| &\leq \delta_{P_{l'}}(p_i, p_j) = \delta_{P_{l'}}(p_{i'}, p_{j'}) - (\delta_{P_{l'}}(p_{i'}, p_i) + \delta_{P_{l'}}(p_{j'}, p_j)) \\
&\leq c_s|p_{i'}p_{j'}| - (|p_i p_j| - (w(T_{i'}) + w(T_{j'}) + |uv|)) \\
&\leq c_s(w(T_{i'}) + w(T_{j'}) + |uv|) - |p_i p_j| + w(T_{i'}) + w(T_{j'}) + |uv| \\
&\leq (c_s + 1)(w(T_{i'}) + w(T_{j'})) + (c_s + 1)|uv| - |p_i p_j| \\
&\leq (c_s + 1)(2|uv|/(c_k - 1)) + (c_s + 1)|uv| - |p_i p_j| \\
&\leq (c_s + 1)(2|uv|/(8(1 + c_s))) + (c_s + 1)|uv| - |p_i p_j| \\
\Rightarrow |p_i p_j| &\leq (\frac{1 + c_s}{2} + \frac{1}{8})|uv| = c_s|uv|.
\end{aligned}$$

3.4 The Approximation Ratio

Let SOL denote the cost of the output of the algorithm for the input set S and let $\rho^* : S \rightarrow \mathbb{R}^+$ denote an optimal assignment for S of cost OPT. We show that $SOL \leq (1.5 - \epsilon)OPT$.

Let $W = w(MST(S))$, $r = w(R)/W$, e_M denote the longest edge in $MST(S)$ and $l = w(e_M)/W$. As shown in [2], $OPT \geq W + w(e_M) = W(1 + l)$. Next we show several upper bounds on the ratio SOL/OPT, corresponding to the four solutions computed during the algorithms and finally conclude that the minimum among them equals at most $(1.5 - \epsilon)$.

Approximation bound for Solution (i). Due to the analysis of the *Hub* algorithm done in [2], we have $SOL \leq W + (w(P_M) + w(e_M))/2 = W(1.5 - r/2 + l/2)$. Therefore,

$$\frac{SOL}{OPT} \leq \frac{W(1.5 - r/2 + l/2)}{W(1 + l)} = \frac{1.5 - (r - l)/2}{1 + l}. \tag{1}$$

Assume $SOL > (1.5 - \epsilon)OPT$, then

$$1.5 - \epsilon < \frac{SOL}{OPT} < \frac{1.5 - (r - l)/2}{1 + l} < 1.5 - r/2,$$

implies $r < 2\epsilon$ and the following corollary follows.

Corollary 8. *One of the following holds,* $SOL \leq (1.5 - \epsilon)OPT$ *or* $r < 2\epsilon$.

The following lemma is crucial for introducing the bounds for Solutions (ii) and (iv).

Lemma 9. *Let* c_s *and the notation* Q_P *be defined as in procedure* f. *Given a constant* δ,

1. *there exist two pairs of points* $(u, v), (y, w)$ *connecting two disjoint sub-paths in* P_M, *each pair with stretch factor greater than* c_s, *that satisfy* $\delta_{P_M}(u, v) \geq \delta$ *and* $\delta_{P_M}(y, w) \geq \delta$; *or*
2. *there exists an edge* $e \in P_M$ *defining* P_{e^l} *and* P_{e^r} *(the two paths of* $P_M \backslash e$, *to the left and to the right of* e, *respectively), such that for every* $(u, v) \in Q_{P_{e^l}} \cup Q_{P_{e^r}}$, $\delta_{P_M}(u, v) < \delta$.

Proof. If the first condition holds, we are done. Otherwise, fix e to be the rightmost edge in P_M. If the second condition does not hold for e, then there exists exactly one pair $(u, v) \in Q_{P_{e^l}}$ with $\delta_{P_M}(u, v) \geq \delta$. Replace e with the consecutive edge to its left in P_M. Continue the process until for every $(u, v) \in Q_{P_{e^l}}$, $\delta_{P_M}(u, v) < \delta$. Note that this condition hold when P_{e^l} contains a single edge. If the process ends with a separating edge e for which there exists a pair $(u, v) \in Q_{P_{e^r}}$ with $\delta_{P_M}(u, v) \geq \delta$, then u is a common endpoint with the preceding edge in the process and there exists a point $w \in P_{e^l}$ such that the two pairs $(w, u), (u, v)$ satisfy the first condition. ☐

Approximation bound for Solution (ii). Consider the value $c_h W$, where $c_h = 20\epsilon$. One of the two conditions of Lemma 9, denoted by L9.1 and L9.2, respectively, must hold for $\delta = c_h W$. We start by assuming that condition L9.1 holds which leads to Lemma 10.

For every $q \in P_M$, we have that the cost of Solution (ii) equals at most $w(P_M) + \max\{|qp_1|, |qp_z|\} + 2w(R)$. Consider the path $\sim f(P_M)$ obtained from P_M after applying the *flatten* procedure f on the sub-paths P_{e^l} and P_{e^r}. Note that the length of this path is at most $w(P_M) - 2c_h W(1 - 1/c_s)$ and it shares

common endpoints with P_M. Let $\tilde{c}$ be the point on $\sim f(P_M)$ closest to its midpoint. The midpoint may lie on an edge of P_M or on a shortcut performed by f. If it lies on a shortcut, we undo it and only one shortcut remains. Thus, the point $\tilde{c}$ is at Euclidean distance at most $\frac{1}{2}[w(P_M) - c_h W(1-\frac{1}{c_s}) + w(e_M)]$, from both endpoints and we have

$$SOL \le W(1 - r + \frac{1}{2}[(1-r) - c_h(1-\frac{1}{c_s}) + l] + 2r) = W[1.5 - \frac{1}{2}(c_h(1-\frac{1}{c_s}) - l - r)].$$

This implies,

$$\frac{SOL}{OPT} \le \frac{1.5 - \frac{1}{2}(c_h(1-\frac{1}{c_s}) - l - r)}{1+l}. \tag{2}$$

Lemma 10. *If condition L9.1 holds for $\delta = c_h W$ then $SOL \le (1.5-\epsilon)OPT$.*

Proof. Assume towards contradiction that $SOL > (1.5-\epsilon)OPT$, then by Corollary 8, $r < 2\epsilon$ and together with equation (2) we receive

$$\begin{aligned} 1.5 - \epsilon < \frac{1.5 - \frac{1}{2}(c_h(1-\frac{1}{c_s}) - l - r)}{1+l} &< 1.5 - \frac{1}{2}(c_h(1-\frac{1}{c_s}) - r) \\ &< 1.5 - \frac{1}{2}(20\epsilon(\frac{1}{5}) - 2\epsilon) < 1.5 - \epsilon \qquad \Rightarrow \quad \text{contradiction.} \end{aligned}$$

□

From now on, assume condition L9.2 holds for $\delta = c_h W$ and let $e \in P_M$ be the edge satisfying the condition. Let $t = (\sum_{(u,v)\in Q_{P_{e^l}}} \delta_{P_M}(u,v) + \sum_{(u,v)\in Q_{P_{e^r}}} \delta_{P_M}(u,v))\ /\ W$, we give an additional bound to the cost of Solution (ii) using the same arguments used for the case where condition L9.1 holds. Let $\tilde{c}$ be the point on $\sim f(P_M)$ closest to its midpoint. Since every pair $(u,v) \in Q_{P_{e^l}} \cup Q_{P_{e^r}}$ satisfies $\delta_{P_M}(u,v) \le c_h W$, the point $\tilde{c}$ is at Euclidean distance at most $\frac{1}{2}[w(P_M) - (t-c_h)\cdot W(1-\frac{1}{c_s}) + w(e_M)]$ from both endpoints, thus

$$\frac{SOL}{OPT} \le \frac{1.5 - \frac{1}{2}((t-c_h)(1-\frac{1}{c_s}) - l - r)}{1+l}. \tag{3}$$

Note that although the analysis for equation 3 considers the path $\sim f(P_M)$, the *flatten* procedure f is performed after Solution (ii) is computed.

In the analysis of Solution (iii) and (iv) we have $w(R) \le (r+t)W$, since applying f on P_{e^l} and P_{e^r} moves portions of the paths connecting points in $Q_{P_{e^l}} \cup Q_{P_{e^r}}$ to R. Consider the iteration of the algorithm for the edge e and a choice of 4 points $p_l, p_{l'}, p_{r'}, p_r \in P_M$ satisfying: p_r is the rightmost point in P_M with $u_r \in T(p_r)$ connected (at any direction) in G_{ρ^*} to a point in $T(p_{l'})$ for $p_{l'}$ to the left of e and, symmetrically, p_l is the leftmost point in P_M with $u_l \in T(p_l)$ connected (at any direction) to a point in $T(p_{r'})$ for $p_{r'}$ to the right of e. Meaning, there is no edge in G_{ρ^*} connecting between a point in $T(p)$ for $p \in P_l \backslash \{p_l\}$ and a point in $T(q)$ for $q \in P_{r'}$ and no edge connecting between a point in $T(p)$ for $p \in P_r \backslash \{p_r\}$ and a point in $T(q)$ for $q \in P_{l'}$. Let x denote the ratio $w(P_x)/W$.

Approximation bound for Solution (iii). Preforming the *Hub* algorithm on P_l and P_r, separately, result in two assignments with a total cost of at most $1.5(W - w(P_x) - w(R)) + w(e_M)/2 + w(e_M)/2$. Directing the path P_x and the trees in R, and assigning all roots in R their assignment, together with adding the two edges, $(u_l, u_{r'})$ for $u_l \in T(p_l), u_{r'} \in T(p_{r'})$ of minimal length and $(u_{l'}, u_r)$ for $u_{l'} \in T(p_{l'}), u_r \in T(p_r)$ of minimal length (or the edge (p_l, p_r) instead if it is cheaper) costs at most $w(P_x) + 2\cdot w(R) + |u_l u_{r'}| + |u_{l'} u_r|$. Overall, we have a total cost of at most $W + \frac{1}{2}(1 - x + (r+t))\cdot W + lW + |u_l u_{r'}| + |u_{l'} u_r|$.

Since there is an edge connecting a point in $T(p_l)$ with a point in $T(p_{r'})$ (in some direction) and an edge connecting a point in $T(p_{l'})$ with a point in $T(p_r)$ in the optimal solution, we have $OPT \ge W + |u_l u_{r'}| + |u_{l'} u_r| - w(e_M)$, hence,

$$\begin{aligned} \frac{SOL}{OPT} &\le \frac{W(1 - l + \frac{1}{2}(1 - x + (r+t)) + 2l) + |u_l u_{r'}| + |u_{l'} u_r|}{W(1-l) + |u_l u_{r'}| + |u_{l'} u_r|} \\ &\le 1 + \frac{W(\frac{1}{2}(1 - x + (r+t)) + 2l)}{W(1-l) + |u_l u_{r'}| + |u_{l'} u_r|} \le 1 + \frac{1}{2}(1 - x + (r+t)) + 2l. \end{aligned} \tag{4}$$

Approximation bound for Solution (iv). Let $\rho_l : P_l \to \mathbb{R}^+$ and $\rho_r : P_r \to \mathbb{R}^+$ denote the assignments obtained by applying the $1D$ *RA* algorithm on P_l and P_r, respectively, with respect to the distance function h_S. Let $\rho' : P_l \cup P_r \to \mathbb{R}^+$ denote the union of the two assignments and let OPT' denote the cost of ρ', i.e., $OPT' = \sum_{v \in P_l \cup P_r} \rho'(v)$.

Claim 1. *$OPT' \leq OPT$.*

Proof. We show that the optimal assignment ρ^* can be adjusted to an assignment $\rho : P_l \cup P_r \to \mathbb{R}^+$, valid for P_l and P_r, separately, with respect to h_S, of the same cost.
We define,

$$\rho(p_l) = \max_{\substack{v \in T(p_i), \\ l \leq i \leq m}} \{\rho^*(v)\}, \qquad \rho(p_r) = \max_{\substack{v \in T(p_i), \\ m+1 \leq i \leq r}} \{\rho^*(v)\},$$

and for every p_j with $j \in \{1, .., l-1\} \cup \{r+1, .., z\}$, $\rho(p_j) = \max_{v \in T(p_j)} \{\rho^*(v)\}$.

Let u and v be two points in the same sub-path, w.l.o.g., P_l, and let $(u = u_1, u_2, ..., u_k = v)$ be the path from u to v in G_{ρ^*}. Consider the sequence $(u = y_1, y_2, ..., y_k = v)$, obtained by replacing every $u_i \in T(P_l)$ with $y_i = r(u_i)$, and every $u_i \in T(P_x \cup P_r)$ with $y_i = p_l$, for $1 \leq i \leq k$. We prove the above sequence forms a path from u to v in the graph induced by ρ with respect to P_l and h_S, and conclude that ρ is valid and $OPT' \leq OPT$. Consider a pair of consecutive nodes in the above sequence, (y_i, y_{i+1}). Note that if $u_i = p_j$ (resp. $u_{i+1} = p_j$) for $1 \leq j < l$, than $u_{i+1} = p_h$ (resp. $u_i = p_h$) for $1 \leq h \leq m$; thus, for every $1 \leq i < k$, by the definition of ρ and h_S we have, $\rho(y_i) \geq \rho^*(u_i) \geq |u_i u_{i+1}| \geq h_S(y_i, y_{i+1})$ and the claim follows. □

Let $\sim g(\rho') : P_{l'} \cup P_{r'} \to \mathbb{R}^+$ be the union of the assignments obtained after applying the transformation g on each of P_l and P_r, separately. The following lemma bounds its cost.

Lemma 11. *$cost(\sim g(\rho')) \leq [c_s + (c_k + 2c_s(c_k + 1))(r + t)]OPT$.*

Proof. Consider applying transformation g on P_l and P_r. By multiplying the range of every point in $P_l \cup P_r$ by c_s we obtain an assignment of cost $c_s \cdot OPT'$. As for the additive part and the second stage, we analyze the cost with respect to $P_{l'}$, while the case for $P_{r'}$ is symmetric. Every $p_j \in P_{l'}$, is responsible for an additional of at most $X_j = c_k \cdot w(T_j) + \delta_{P_{l'}}(p_{j^-}, p_{j^+}) \leq c_k \cdot w(T_j) + c_s|p_{j^-} p_{j^+}| = c_k \cdot w(T_j) + c_s[2c_k \cdot w(T_j) + w(T_{j^-}) + w(T_{j^+})]$ to the total cost, where p_{j^-} and p_{j^+} are defined as in the definition of g. The first element in the summation is the range added to p_j itself, and the second is the cost of directing the path between p_{j^-} and p_{j^+} towards p_j (depicted in Fig. 5 in black).

Allegedly, for computing $cost(\sim g(\rho'))$ we should sum X_j over all $p_j \in P_{l'} \cup P_{r'}$ and add it to the cost $c_s \cdot OPT'$, however, we observe that it is suffices to consider a point p_i only once as p_{j^-}, for the rightmost point p_j such that $i = j^-$ and only once as p_{j^+}, for the leftmost point p_k such that $i = k^+$. Thus, we can charge p_{j^-} and p_{j^+} themselves once on each of the elements $c_s \cdot w(T_{j^-})$ and $c_s \cdot w(T_{j^+})$ in the overall summation. Namely, charge every point p_j for a total range increase of $Y_j = w(T_j)[c_k + c_s(2c_k + 2)]$. Summing Y_j over all $p_j \in P_{l'} \cup P_{r'}$ and adding the cost $c_s \cdot OPT'$, using Claim 1 gives

$$\begin{aligned} cost(g(rho')) &\leq c_s \cdot OPT' + \sum_{1 \leq j \leq z} w(T_j)[c_k + c_s(2c_k + 2)] \\ &= c_s \cdot OPT' + [c_k + 2c_s(c_k + 1)](w(R)) \\ &\leq [c_s + (c_k + 2c_s(c_k + 1))(r + t)]OPT. \end{aligned}$$

□

Note that g has already assigned to every $p_j \in P_{l'} \cup P_{r'}$ an assignment greater than $w(T(p_j))$. Directing all trees in R towards their roots, directing the path P_x and adding the edge (p_l, p_r), adds to the cost at most $(2x + (r + t))W < (2x + (r + t))OPT$, and together with Lemma 11 we receive

$$\frac{SOL}{OPT} \leq c_s + (c_k + 2c_s(c_k + 1) + 1)(r + t) + 2x \tag{5}$$

Lemma 12. *If condition L9.2 holds for $\delta = c_h W$, then $SOL \leq (1.5 - \epsilon)OPT$.*

Proof. Assume towards contradiction that $SOL > (1.5-\epsilon)OPT$. By Corollary 8, $r < 2\epsilon$ and together with equation (3) we receive,

$$1.5-\epsilon < \frac{1.5-\frac{1}{2}((t-c_h)(1-\frac{1}{c_s})-l-r)}{1+l} < 1.5-\frac{1}{2}((t-20\epsilon)\frac{1}{5}-r)$$
$$< 1.5-\frac{t}{10}+3\epsilon \qquad \Rightarrow \quad t < 40\epsilon.$$

Replacing r and t with the above upper bounds in equation (5) gives,

$$1.5-\epsilon < c_s + (c_k + 2c_s(c_k+1)+1)(r+t)+2x$$
$$< 1\frac{1}{4} + 70(42\epsilon)+2x \qquad \Rightarrow \quad x > \frac{1}{8} - 1472\epsilon,$$

and by equation (4) we have,

$$1.5-\epsilon < 1+\frac{1}{2}(1-x+(r+t))+2l < 1+\frac{1}{2}(1-(\frac{1}{8}-1472\epsilon)+42\epsilon)+2l$$
$$< 1.5-\frac{1}{16}+757\epsilon+2l \qquad \Rightarrow \quad l > \frac{1}{32}-380\epsilon.$$

The upper bound on l together with equation (1) imply,

$$1.5-\epsilon < \frac{1.5-(r-l)/2}{1+l} < \frac{1}{2}+\frac{1}{1+l} < \frac{1}{2}+\frac{1}{1+\frac{1}{32}-380\epsilon} \qquad \Rightarrow \quad \epsilon > \frac{8}{10^5}$$

in contradiction to our choice of $\epsilon = \frac{5}{10^5}$. $\square$

We conclude with Theorem 13, derived from Lemma 9 together with Lemmas 10 and 12.

Theorem 13. *Given a set S of points in $\mathbb{R}^d$ for $d \geq 2$, a minimum cost range assignment $(1.5-\epsilon)$-approximation can be computed in polynomial time for S, where $\epsilon = \frac{5}{10^5}$.*

The reader can notice that our algorithm yields a better approximation bound than stated in the above theorem. However, we preferred the simplicity of presentation over a more complicated analysis resulting in a tighter bound.

References

[1] K. Abu-Affash, R. Aschner, P. Carmi, and M. J. Katz. Minimum power energy spanners in wireless ad-hoc networks. In *INFOCOM*, 2010.

[2] C. Ambühl, A. E. F. Clementi, P. Penna, G. Rossi, and R. Silvestri. On the approximability of the range assignment problem on radio networks in presence of selfish agents. *Theor. Comput. Sci.*, 343(1-2):27–41, October 2005.

[3] A. E. F. Clementi, P. Penna, and R. Silvestri. On the power assignment problem in radio networks. *Mob. Netw. Appl.*, 9(2):125–140, April 2004.

[4] A.E.F. Clementi, A. Ferreira, P. Penna, S. Perennes, and R. Silvestri. The minimum range assignment problem on linear radio networks. In *ESA*, 2000.

[5] G. K. Das, S. C. Ghosh, and S. C. Nandy. Improved algorithm for minimum cost range assignment problem for linear radio networks. *Int. J. Found. Comput. Sci.*, 18(3):619–635, 2007.

[6] L. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243(1-2):289–305, 2000.

[7] K. Pahlavan. *Wireless information networks.* John Wiley, Hoboken, NJ, 2005.

[8] H. Shpungin and M. Segal. Near optimal multicriteria spanner constructions in wireless ad-hoc networks. In *INFOCOM*, pages 163–171, 2009.

[9] Y. Wang and X.-Y. Li. Distributed spanner with bounded degree for wireless ad hoc networks. In *IPDPS '2002:*, page 120, 2002.

[10] Y. Wang and X.-Y. Li. Minimum power assignment in wireless ad hoc networks with spanner property. *Journal of Combinatorial Optimization*, 11(1):99–112, 2006.